IN THE CLAIMS

Please amend the claims as follows:

1. (Currently Amended) A method for scheduling ~~program units~~ streams of instructions, the method comprising:

starting a process within an operating system of a processor;

starting at least one thread within the operating system, the at least one thread associated with the process;

executing a plurality of streams of instructions within the at least one thread;

entering a kernel mode by a first stream of instructions of the plurality of streams of instructions upon the occurrence of a context shifting event; and

if the first stream entering the kernel mode must ~~block~~ be blocked, then blocking the execution of the ~~other streams of the plurality of streams~~ others of the plurality of streams of instructions subsequent to the first stream of instructions.

2. (Currently Amended) The method of claim 1, further comprising saving [[the]] a context of each of the plurality of streams of instructions in a thread context data structure.

3. (Currently Amended) The method of claim 2, wherein each one of the streams of instructions are executed on a separate processor.

4. (Original) The method of claim 1, wherein the context shifting event comprises an exception.

5. (Original) The method of claim 4 wherein the exception comprises a signal.

6. (Original) The method of claim 1 wherein the context shifting event comprises a non-local goto.

7. (Original) The method of claim 1, wherein the context shifting event comprises a system call.

8. (Currently Amended) A system for scheduling streams of instructions, the system comprising:

at least one multiple processor unit having a plurality of processors;

a memory coupled to the plurality of processors; and

an application executed by at least one of the plurality of processors ~~from the memory and operable~~ to perform the ~~tasks~~ steps of:

starting a process within an operating system of at least one of the plurality of processors,

starting at least one thread within the operating system, the at least one thread associated with the process;

executing a plurality of streams of instructions within the at least one thread,

entering a kernel mode by a first stream of instructions of the plurality of streams of instructions upon the occurrence of a context shifting event, and

if the first stream of instructions entering the kernel mode must ~~block~~ be blocked, then blocking the execution of the ~~other streams~~ others of the plurality of streams of instructions of subsequent to the first stream of instructions.

9. (Currently Amended) The system of claim 8, further comprising saving [[the]] a context of each of the plurality of streams of instructions in a thread context data structure.

10. (Currently Amended) The system of claim 9, wherein each one of the ~~streams~~ stream of instructions of the plurality of streams of instructions are executed on a separate processor ~~of the multiple processor unit~~.

11. (Original) The system of claim 8, wherein the context shifting event comprises an exception.

12. (Original) The system of claim 11 wherein the exception comprises a signal.

13. (Original) The system of claim 8 wherein the context shifting event comprises a non-local goto.

14. (Original) The system of claim 8, wherein the context shifting event comprises a system call.

15. (Currently Amended) A computer-readable media having computer executable ~~instructions for performing~~ codes executing by a processor that perform a method [[for]] of scheduling ~~program units, the method~~ streams of instructions comprising:

starting a process within an operating system of a processor;

starting at least one thread within the operating system, the at least one thread associated with the process;

executing a plurality of streams of instructions within the at least one thread;

entering a kernel mode by a first stream of instructions of the plurality of streams of instructions upon the occurrence of a context shifting event; and

if the first stream of instructions entering the kernel mode must ~~block~~ be blocked, then blocking the execution of the ~~other streams~~ others of the plurality of streams of instructions subsequent to the first stream of instructions.

16. (Currently Amended) The computer-readable media having computer executable codes executing by a processor that perform a method of scheduling streams of instructions of claim 15, further comprising saving [[the]] a context of each of the plurality of streams of instructions in a thread context data structure.

17. (Currently Amended) The computer-readable media of claim 16, wherein each one of the ~~stream are~~ streams of instructions is executed on a separate processor.

18. (Original) The computer-readable media of claim 15, wherein the context shifting event comprises an exception.

19. (Original) The computer-readable media of claim 18 wherein the exception comprises a signal.

20. (Original) The computer-readable media of claim 15 wherein the context shifting event comprises a non-local goto.

21. (Original) The computer-readable media of claim 15, wherein the context shifting event comprises a system call.